TEXAS INSTRUMENTS

Interfacing the TLC5618A Digital-to-Analog Converter to the TMS320C203 DSP

Application Report

1998

Advanced Analog Products

Texas Instruments

Printed in U.S.A.
07/98

SLAA033

Interfacing the TLC5618A Digital-to-Analog Converter to the TMS320C203 DSP Application Report

Literature Number: SLAA033
July 1998

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

CERTAIN APPLICATIONS USING SEMICONDUCTOR PRODUCTS MAY INVOLVE POTENTIAL RISKS OF DEATH, PERSONAL INJURY, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE ("CRITICAL APPLICATIONS"). TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF TI PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty or endorsement thereof.

Contents

List of Figures

List of Tables

Interfacing the TLC5618A Digital-to-Analog Converter to the TMS320C203 DSP

ABSTRACT

This application report describes the hardware and software needed to interface the TLC5618A dual 12-bit digital-to-analog converter (DAC) to the TMS320C203 digital signal processor (DSP). The solution presented provides the bridge between the system digital signals and the analog signal processing. The software organization and flow chart are presented with the software listing.

1 Introduction

The interface between the digital and analog domains can present a difficult design problem. Hardware and software must work together to produce a usable design. This application report offers a design solution for interfacing the TLC5618A dual 12-bit digital-to-analog converter (DAC) and TMS320C203 digital signal processor (DSP). The report describes the hardware and software needed to bridge the gap between the system digital signals and the analog signal processing. The body of the report discusses the basic operation of the TLC5618A, the DSP/DAC interface, and the basics of the DSP serial port operation. The last section contains the software flow chart and program listing.

The software application describes the buffer loading, buffer and B register loading, simultaneous A and B channel updates, and power-down function. The software and hardware reproduce memory-stored triangle and sine waveforms at the output of the A and B DAC channels.

Figure 1 shows the hardware application schematic.

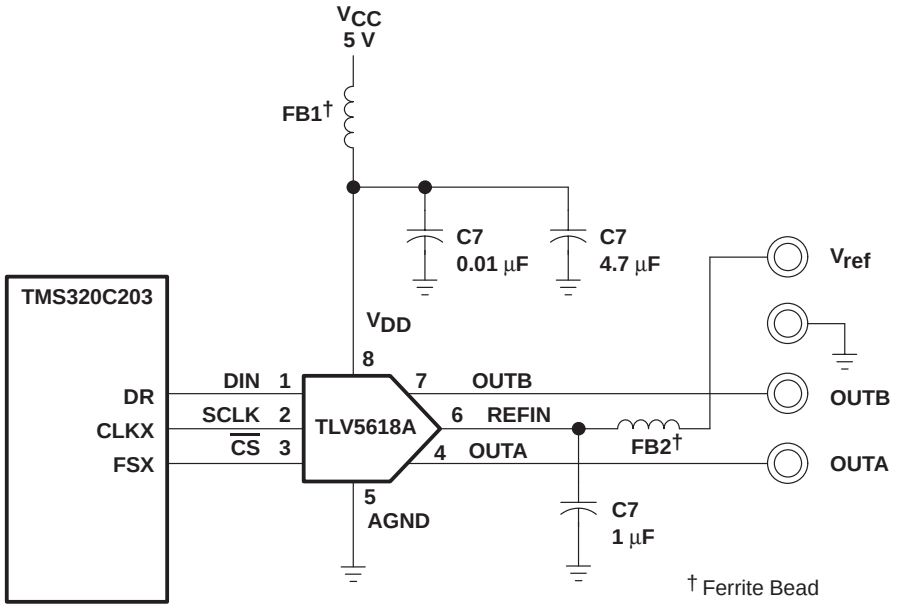


Figure 1. Application Schematic

2 TLC5618A Overview

The TLC5618A is a CMOS 12-bit serial-in DAC using standard digital switching ladder networks in an 8-pin package. Figure 2 shows the package pin out, and Table 1 lists and describes the pins and I/O signals. Figure 3 shows the functional block diagram. The device has three digital control signal pins: chip-select (CS), input clock (SCLK), and data input (DIN); and two analog outputs (OUT A and OUT B). Additionally, a reference input pin (REFIN) establishes the output voltage range at two times V(REFIN). The DSP transmit clock, CLKX, applied at SCLK, clocks data in on the falling edge.

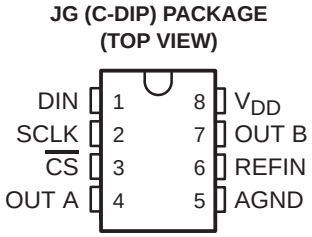


Figure 2. Package Diagram

Table 1. Terminal Functions

TERMINAL NAME	TERMINAL NO.	I/O	DESCRIPTION
DIN	1	I	Serial data input
SCLK	2	I	Serial clock input
$\overline{CS}$	3	I	Chip select, active low
OUT A	4	O	DAC A analog output
AGND	5		Analog ground
REFIN	6	I	Reference voltage input
OUT B	7	O	DAC B analog output
V_{DD}	8		Positive power supply

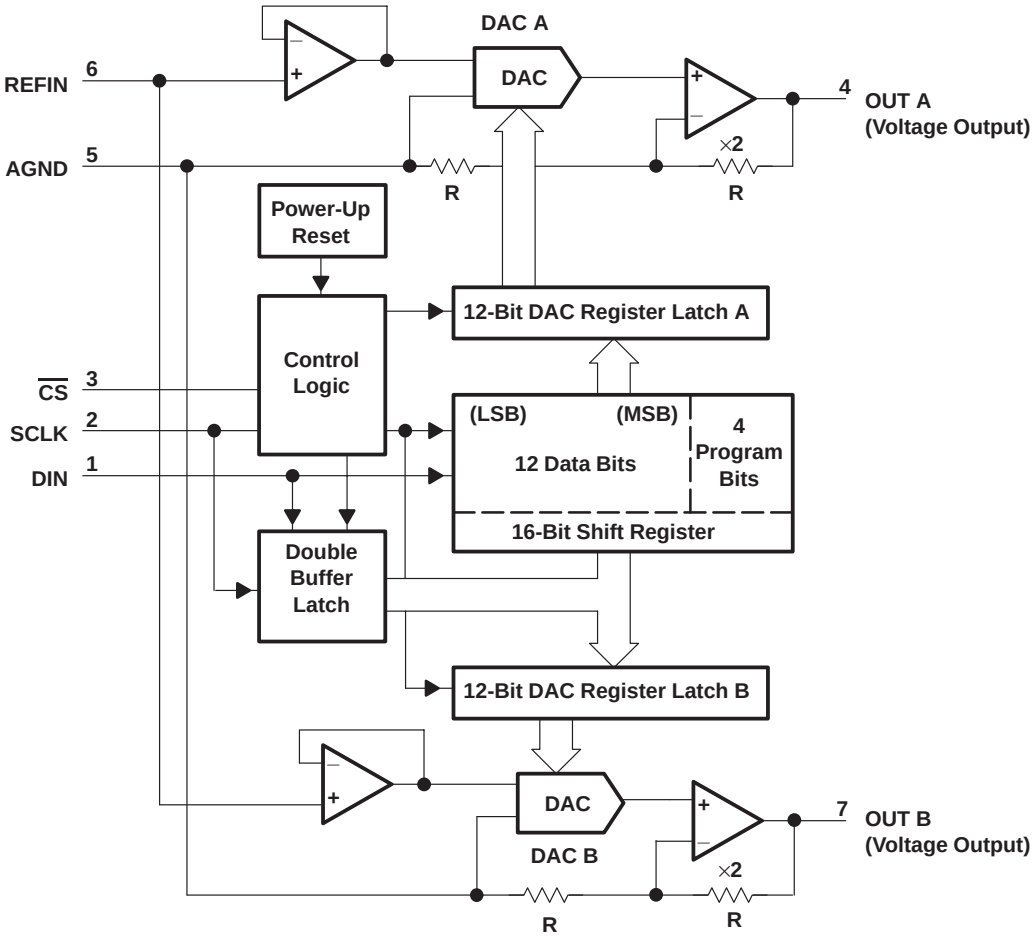


Figure 3. Functional Block Diagram

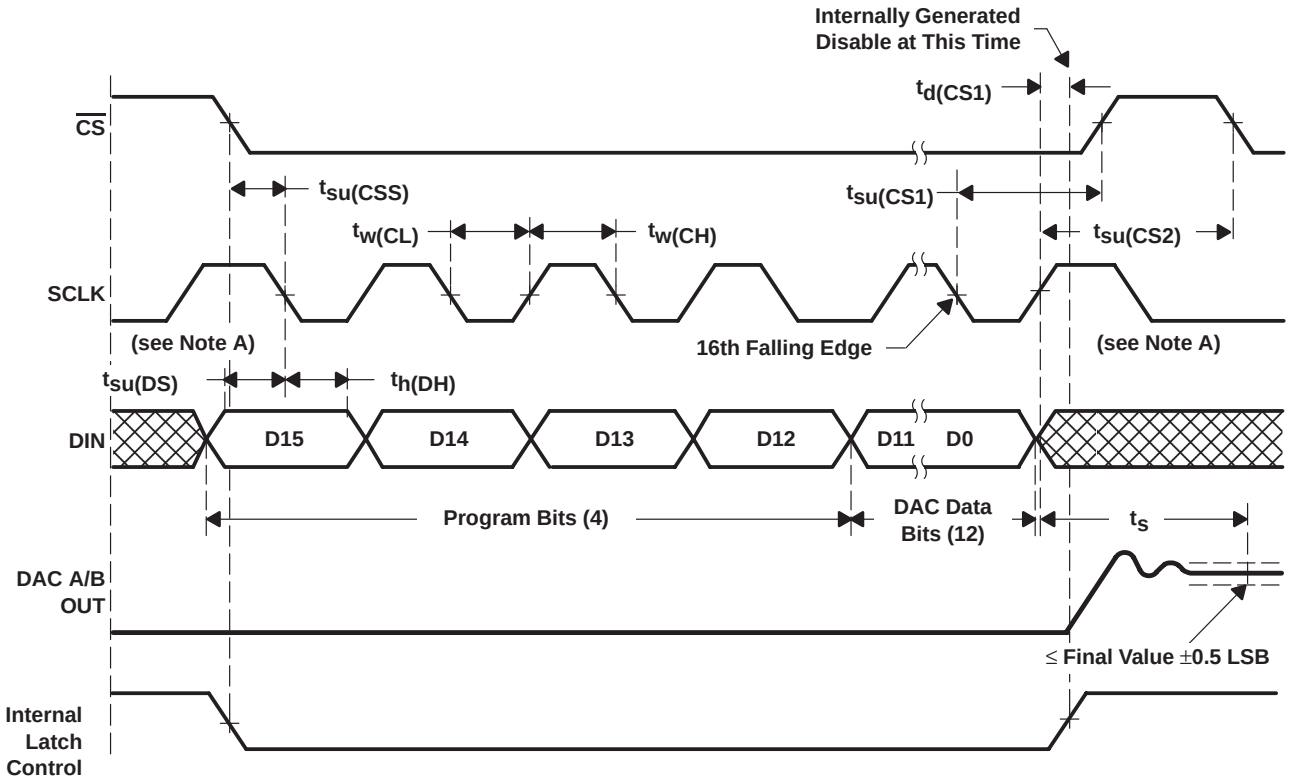
3 Operational Overview

The following paragraphs describe the overall operational function of the TLC5618A.

3.1 Signal Sequence

As shown in the timing sequence in Figure 4, a high level on the $\overline{CS}$ pin disables the DIN and SCLK and takes DATA OUT to a high impedance state. When taken low, $\overline{CS}$ enables the device inputs, but no data is transferred until the falling edge of FSX is received from the DSP to $\overline{CS}$.

After the falling edge of the DSP FSX, the TLC5618 recognizes the data clock, CLKX, at SCLK and receives the program input data at DIN on the first four falling edges of the data clock. These four program bits configure the DAC for fast/slow mode, register loading option, and power down selection. The remaining 12 bits are the DAC data.



NOTE A: The input clock, applied at the SCLK terminal, should be inhibited high when $\overline{CS}$ is high to minimize clock feedthrough.

Figure 4. Timing Diagram for TLC5618A Only

3.2 Reference Voltage Inputs

Twice the voltage difference between the REFIN and the ground terminal, AGND, determines the analog output range, that is, the upper and lower limits of the analog outputs at the full-scale and zero-scale readings, respectively. The voltage values applied to REFIN and the analog outputs with respect to the AGND terminal must not be higher than the positive supply or lower than ground. The analog voltage output is full scale when the digital input is equal to all 1s and equal to zero when the input is equal to all 0s.

3.3 Input Data Bits

DIN is internally connected to a 4-bit serial input data register. The input data for the DAC are control data which select different modes of operation. The DSP provides the data word with the MSB first. Each data bit clocks in on the falling edge of the SCLK sequence. Table 2 shows the software program modes.

Table 2. Program Modes

PROGRAM BITS				DEVICE FUNCTION
D15	D14	D13	D12	
1	X	X	X	Write to latch A with serial interface register data and latch B updated with buffer latch data
0	X	X	0	Write to latch B and double buffer latch
0	X	X	1	Write to double buffer latch only
X	0	X	X	12.5 μs settling time
X	1	X	X	2.5 μs settling time
X	X	0	X	Powered-up operation
X	X	1	X	Power down mode

4 The DAC/DSP System

The software configures the DSP serial port to the 16-bit master mode so that the DSP generates the frame sync signal at FSX and the data clock at CLKX serial port terminals.

From the hardware schematic the analog outputs are OUT A and OUT B. Table 3 lists the digital signal connections between the DSP and the DAC.

Table 3. DSP/DAC Interconnection

FROM DSP	TO DSP	TO DAC
CLKX	CLKR	SCLK
DX		DIN
FSX		CS

The following statements describe the generation and application of the configuration and control signals. (See Figure 1 and Figure 3).

1. The DSP CLKX output provides a 20-MHz data clock which is a divide-by-2 of the DSP master clock.
2. The DSP DX output supplies the 16-bit control to the TLC5618A at DIN.
3. The falling edge of the frame sync signal (FSX) of the DSP serial port initiates the data transfer between the DSP and the DAC.
4. An internal counter of the TLC5618A provides a register latch command on the 17th rising edge of CLKX signal to the SCLK input.

5 The DSP Serial Port

The serial port provides direct communication with serial I/O devices and consists of six basic signals and five registers. The DSP Internal Serial Port Operation section discusses the registers. The six signals are:

- CLKX
  Serial transmit clock. This signal clocks the transmitted data from the DX terminal to the DIN terminal of the TLC5618A.
- CLKR
  Serial receive clock. This signal clocks data into the DSP DR terminal, but is not required in this application.
- DX
  Data transmit. From this terminal the DSP transmits 16-bit data to the DIN terminal of the TLC5618A.
- DR
  Data receive. The DSP receives 16-bit data from the peripheral device, but is not necessary in this application.
- FSX
  Frame sync transmit. This signal frames the transmit data. The DSP begins to transmit data from DX on the falling edge of FSX and continues to transmit data for the next 16 clock cycles from the CLKX terminal. The FSX signal is applied to the TLC5618A $\overline{CS}$ terminal..
- FSR
  Frame sync receive. This signal frames the receive data. From a peripheral device, the DSP begins to receive data on the falling edge of FSR and continues to recognize valid data for the following 16 clocks from CLKR. This function is not necessary for this application.

Table 4 lists and describes the serial port pins and registers.

Table 4. DSP Serial Port Signals and Registers

PINS	DESCRIPTION	REGISTERS	DESCRIPTION
CLKX	Transmit clock signal	SSPCR	Synchronous serial port control register
CLKR	Receive clock signal	SDTR	Synchronous data transmit and receive register
DX	Transmitted serial data signal	XSR	Transmit shift register
DR	Received serial data signal	RSR	Receive shift register
FSX	Transmit frame synchronization signal		
FSR	Receive frame synchronization signal		

For this application the DSP serial port is programmed as the master, so the CLKX output is fed to the CLKR terminal and the FSX output is fed to the FSR terminal.

5.1 DSP Internal Serial Port Operation

Three signals are necessary to connect the transmit pins of the transmitting device with the receive pins of the receiving device for data transmission:

- DX, the transmitted serial data signal, sends the actual data.
- FSX initiates the transfer (at the beginning of the 16-bit packet).
- CLKX clocks the bit transfer.

The corresponding pins on the receive device are DR, FSR, and CLKR, respectively.

As shown in Figure 5, the transmit data is written to SDTR transmit, and received data is read from SDTR receive. A transmit command is executed by writing data to the SDTR transmit FIFO buffer, which copies the data to the XSR when the XSR is empty.

The XSR manages the shifting of the data to the DX pin, thus allowing another write to SDTR transmit as soon as the SDTR-to-XSR copy is completed. On completion of the SDTR-to-XSR copy, a 0-to-1 transition occurs on the XRDY bit in the SSPCR and generates an XINT.

The process is similar for receiving. On completion of the RSR-to-SDTR copy, a 0-to-1 transition occurs on the RRDY bit in the SSPCR and generates a RINT.

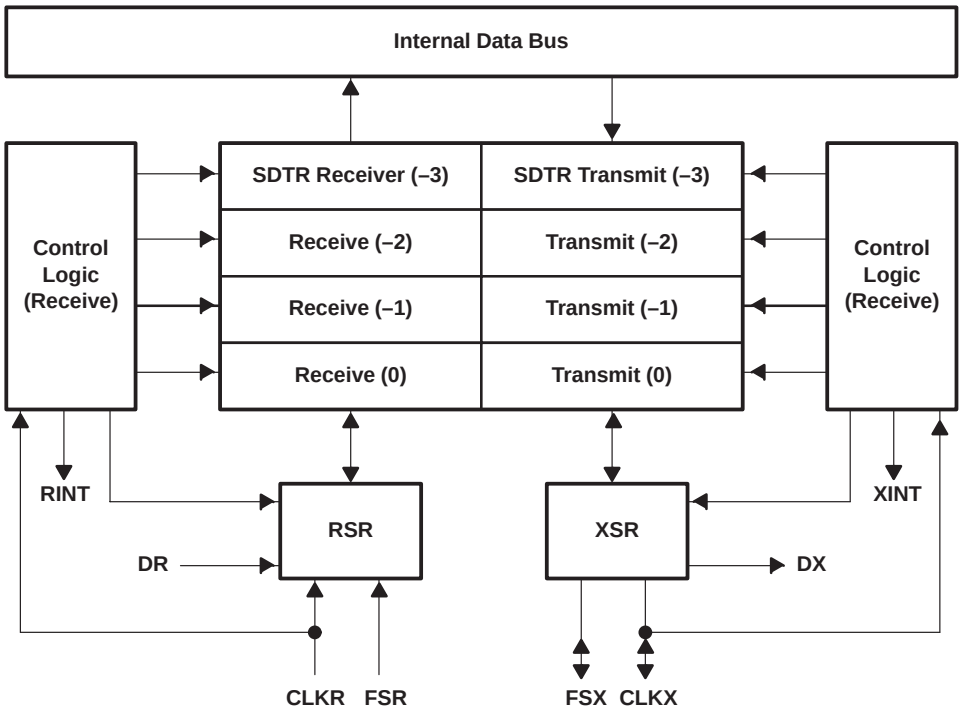


Figure 5. Internal DSP Serial Port Block Diagram

6 Software Overview

This interface program generates a triangle waveform from DSP to TLC5618A. Figure 6 shows the program flowchart.

The C203 CLKINX2 frequency is 80 MHz, the CPU CLK for the C203 DSP is 40 MHz, SCLK for the TLC5618A is driven by CLKX, which is 20 MHz.

The program starts with a common initialization procedure for the DSP followed by the initialization of the serial port.

The following steps initialize the DSP.

1. Disable the global interrupts.
2. Set the data page pointer to 0h.

The next two steps initialize the serial port.

3. Set the synchronous serial port control register to 0C00Eh (SSPCR=0C00Eh)

   The individual bits within the SSPCR now contain the following functions:

   The frame sync mode bit is set to 1 (FSM=1) to allow burst mode operation.

   The clock mode bit is set to 1 (MCM=1) to set the transmit clock CLKX to1/2 of the DSP master clock of 40 MHz (CLKX frequency = 20 MHz)

   The transmit mode bit is set to 1 (TXM=1) to generate the frame sync internally as required for the data transfer initiation. FSX is now programmed as an output.

   The FREE and SOFT bits are set to 1. The clock continues to run at breakpoint.
4. Set the synchronous serial port control register to 0C03Eh (SSPCR=0C03Eh)

   Writing 1s to the transmitter reset field (XRST) and to the receiver reset field (RRST) activates the serial port transmitter and receiver.

The program executes the following steps:

1. Initializes the C203 DSP
2. Initializes the serial port
3. DSP generates a triangle waveform to TLC5618A.
4. After obtaining all waveform(AR7=0), the DSP exits the `triangle_wave` subroutine.

The data is now available for use in user defined functions (algorithms).

6.1 Flowchart

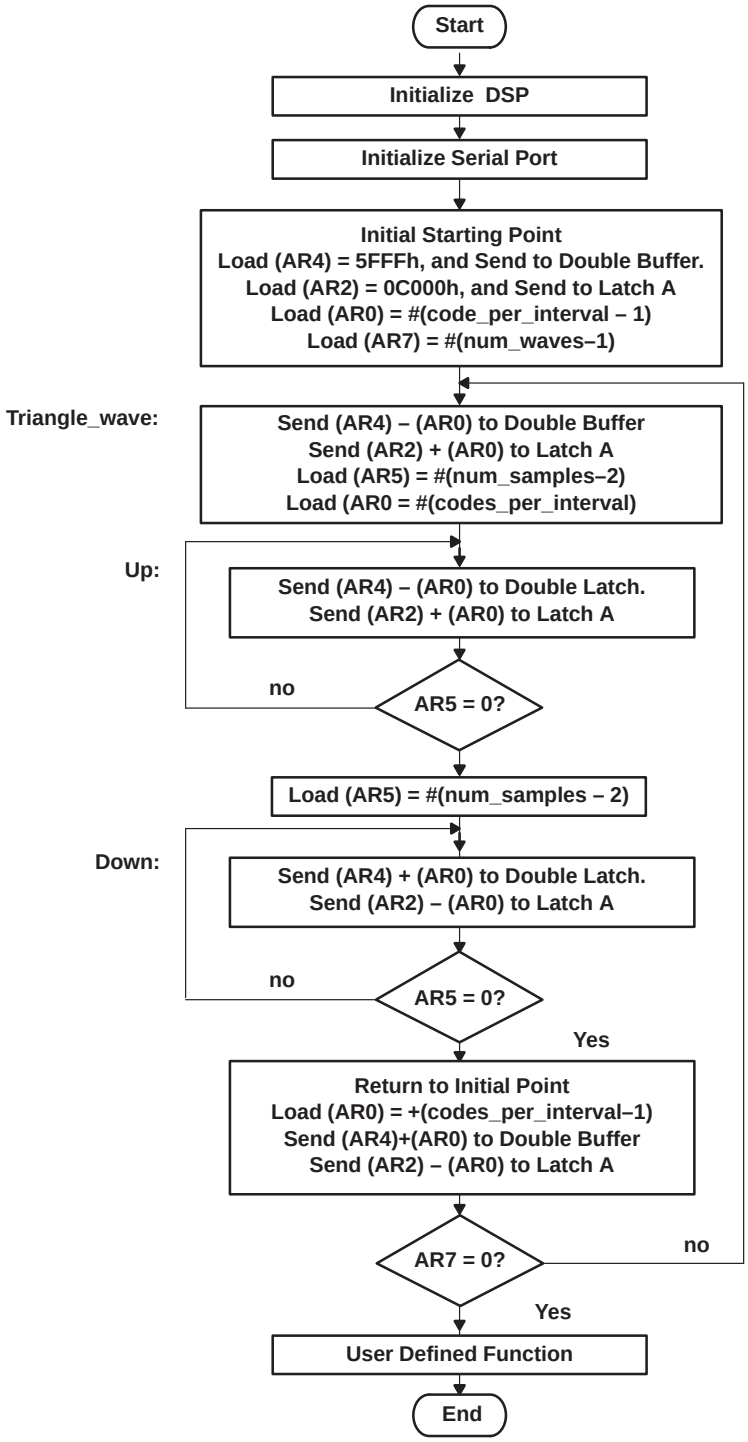


Figure 6. TLC5618A to TMS320C203 80 MHz DSP Interface Program Flow Chart

6.2 Program Listing

```
; ***********************************************************************
; *       (C) COPYRIGHT TEXAS INSTRUMENTS, INC. 1998            *
; ***********************************************************************
;
***********************************************************************
; *                                                             *
; *       File: AC5618DA.ASM  Main routine                      *
; *                                                             *
; *       Written By: David Quach                               *
;
***********************************************************************
          .title ”TLV5618A DAC Interface routine”
;
***********************************************************************
; This routine allows the ’C2O3EVM to interface with a DAC on the
; Serial port of the DSP.
;
;      Program Bits D15 – D12 Function
------------------------------------------------------------------------
;  D15    D14    D13    D12 |
;    1                    | Write to latch A from DSP and latch B
;                         | updated with buffer latch data.
;    0                 0  | Write to latch B and double buffer.
;    0                 1  | Write to double buffer latch only.
;          0              | 12.5 µs setting time.
;          1              | 2.5 µs setting time.
;                0        | Powerer–up operation.
;                1        | Power down mode.
;------------------------------------------------------------------------
;
             .mmregs
SDTR         .set   0fff0h
SSPCR        .set   0fff1h
Latch_A      .set   060h
Latch_B      .set   061h
Double_Buffer.set   062h
temp0        .set   063h
       .     .data
num_samples  .set   8         ; (Number of intervals per cycle) / 2
codes_per_interval .set 512   ; (4096 counts) / (num_samples)
num_waves    .set   10        ; Number of triangle wave
```

```
        .ps    1000h            ; Starting Program Address = 1000h
        .text
start:
        SETC   INTM             ; disable global interrupts
        LDP    #0               ; set data page pointer
        CLRC   INTM
; Initialize Serial Port
        SPLK   #0c00eh,temp0    ; FREE=SOFT=TXM=MCM=FSM=1
        OUT    temp0, SSPCR
        SPLK   #0c03eh, temp0   ; XRST=RRST=1 reset the receiver and
; transmitter interface
        OUT    temp0, SSPCR
        lar    AR7,#num_waves–1 ; (AR7) = #(num_waves–1)


; DSP will generate a triangle wave to DAC Latch A and double buffer,
; while ; latch A is beginning from 0h, increment by 200h (512)
; to FFFh (4095). Then  decrease to 0h, decrement of 200h (512).
; The process is opposite for double buffer. It begins from FFFh (4095)
; decrement by 200h (512), to 0h. Then increase to FFFh (4095)
; increment of 200h (512)
* Initial Start Point
        lar    AR4, #5FFFh      ; (AR4) = #5FFFh
        sar    AR4, Double_Buffer  ; write full scale to double buffer
        LAR    AR2, #0C000h     ; (AR2) = #0C000h
        sar    AR2, Latch_A     ; Write 0 to latch A
* Send the initial data to DAC
        out    Double_Buffer, SDTR
        rpt    #99              ; 2.5 µs/25 ns = 100 at CLKOUT1=40 MHz
        nop                     ; Wait 2.5 ms for output settling time
        out    Latch_A, SDTR
        rpt    #99              ; 2.5 µs/25 ns = 100 at CLKOUT1=40MHz
        nop                     ; Wait 2.5 ms for output settling time
        LAR    AR0, #codes_per_interval–1 ;AR0=#(codes/interval–1) = 511
Triangle_wave:


    mar       *, AR4            ; Select AR4
    mar       *0–               ; (AR4) = (AR4)–(AR0) = (5E00h)
    sar       AR4, Double_Buffer  ; Send (AR4) to double buffer
    out       Double_Buffer, SDTR
    rpt       #99               ; 2.5 µs/25 ns = 100 ;CLKOUT1=40 MHz
    nop                         ; wait 2.5 µs for output settling time
    mar       *, AR2            ; Select AR2
    mar       *0+               ; (AR2) = (AR2) + (AR0) = (C1FFh)
    sar       AR2, Latch_A      ; Send (AR2) to latch A
    out       Latch_A, SDTR
```

```
    rpt     #99             ; 2.5 µs/25 ns = 100 ;CLKOUT1=40MHz
    nop
    LAR     AR5, #num_samples–2       ; (AR5) = #(num_samples–2)
    LAR     AR0, #codes_per_interval  ; (AR0)= codes/interval
up:
    mar     *, AR4          ; select AR4
    mar     *0–             ; (AR4) = (AR4)–(AR0)
    sar     AR4, Double_Buffer ; write (AR4) to double buffer
    out     Double_Buffer, SDTR
    rpt     #99             ; 2.5 µs/25 ns = 100 CLKOUT1=40 MHz
    nop
    mar     *, AR2          ; select AR2
    mar     *0+             ; (AR2)=(AR2)+(AR0)
    sar     AR2, Latch_A    ; write (AR2) to latch A
    out     Latch_A, SDTR
    rpt     #99             ; 2.5 µs/25 ns = 100 CLKOUT1=40 MHz
    nop
    mar     *, AR5          ; select AR5
    BANZ    up              ; IF AR5 <>0, GO BACK TO UP, AND (AR5)–1
    LAR     AR5, #num_samples–2 ;(AR5)=#(num_samples–2)
down:
    mar     *, AR4          ; select AR4
    mar     *0+             ; (AR4) = (AR4) + (AR0)
    sar     AR4, Double_Buffer ; write (AR4) to double buffer
    out     Double_Buffer, SDTR
    rpt     #99             ; 2.5 µs/25 ns = 100 CLKOUT1=40 MHz
    nop
    mar     *, AR2          ; select AR2
    mar     *0–             ; (AR2) = (AR2) – (AR0)
    sar     AR2, Latch_A    ; write (AR2) to latch A
    out     Latch_A, SDTR
    rpt     #99             ; 2.5 µs/25 ns = 100 CLKOUT1=40 MHz
    nop
    mar     *, AR5          ; select AR5
    banz    down            ; IF AR5<>0, go back TO down, and
                            ; (AR5) = (AR5)–1
* Last step; Return to initial point
    LAR     AR0, #codes_per_interval–1  ;AR0=(codes/interval)–1 = 511
```

```
   mar       *, AR4           ; select AR4
   mar       *0+              ; (AR4)=(AR4)+(AR0) = (5FFFh)
   sar       AR4, Double_Buffer ; write (AR4) to double buffer
   out       Double_Buffer, SDTR
   rpt       #99              ; 2.5 µs/25 ns = 100 CLKOUT1=40 MHz
   nop
   mar       *, AR2           ; select AR2
   mar       *0–              ; (AR2)=(AR2)–(AR0) = (C000h)
   sar       AR2, Latch_A     ; write (AR2) to latch A
   out       Latch_A, SDTR
   rpt       #99              ; 2.5 µs/25 ns = 100 CLKOUT1=40 MHz
   nop
   mar    *, AR7              ; select AR7
   banz      Triangle_wave    ; if AR7<>0,go back to Triangle_wave,
                              ; (AR7)=(AR7)–1
   splk      #2000h, temp0    ; send power down to DAC
   out       temp0, SDTR
;** User Defined
;**    FUNCTION(S)
;**
end_loop:
   NOP
   NOP
   B         end_loop
   .end
```

7 Summary

This application report describes the hardware and software needed to interface the TLC5618A dual 12-bit digital-to-analog converter (DAC) to the TMS320C203 digital signal processor (DSP). The solution presented provides the bridge between the system digital signals and the analog signal processing.